# **TestNG Framework**

# **TestNG Introduction**

TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionality that make it more powerful and easier to use.

TestNG is an open source automated testing framework; where NG of TestNG means NextGeneration. TestNG is similar to JUnit but it is much more powerful than JUnit but still it’s inspired by JUnit. It is designed to be better than JUnit, especially when testing integrated classes.

TestNG eliminates most of the limitations of the older framework and gives the developer the ability to write more flexible and powerful tests with help of easy **annotations, grouping, sequencing & parameterizing.**

## **Benefits of TestNG**

There are number of benefits of TestNG but from Selenium perspective, major advantages of TestNG are :

**1) It gives the ability to produce HTML Reports of execution**

**2) Annotations made testers life easy.**

**3) Test cases can be Grouped and Prioritized more easily.**

**4) Parallel testing is possible (Test Suite Execution/Bulk Test Execution)**

**5) Generates Logs.**

**6) Data Parameterization is possible/Data Driven Test.**

## **Annotations in TestNG**

@AfterTest: The annotated method will be run after all the test methods belonging to the classes inside the tag have run.

@BeforeSuite: The annotated method will be run before all tests in this suite have run.

@AfterSuite: The annotated method will be run after all tests in this suite have run.

@BeforeTest: The annotated method will be run before any test method belonging to the classes inside the tag is run.

ups: The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked.

@AfterGroups: The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked.

@BeforeClass: The annotated method will be run before the first test method in the current class is invoked.

@AfterClass: The annotated method will be run after all the test methods in the current class have been run.

@BeforeMethod: The annotated method will be run before each test method.

@AfterMethod: The annotated method will be run after each test method.

@Test: The annotated method is a part of a test case.

**Install the TestNG by following the below steps:**

1. Goto Help-> Install New Software in Eclipse IDE.
2. Click on Add button.
3. Enter Name as “TestNG”
4. Enter location as <http://beust.com/eclipse>
5. Click on OK button
6. Select “**TestNG**” software and click Next button.
7. Click Next and Accept the “Terms and Conditions”
8. Click on Finish.

**To Create a new TestNG Class follow the below steps:**

1. Right Click on the Project-> Select TestNG-> Create TestNG Class..
2. Select the Src and Package Folder Names.
3. Enter a Valid Class Name.
4. Select the required Annotations like @Test, @BeforeMethod, @AfterMethod etc.

**Exmple of TestNG using the @Test Annotation to understand the order, Priority, Description and Enabled attributes**

package testPackage1;

import org.testng.Assert;

import org.testng.annotations.Test;

public class TestNG {

@Test (priority = 1, description = "test by shanker", enabled=false)

public void Test1() {

System.out.println("TestF");

Assert.assertEquals("Exp", "Exp");

}

@Test (priority = 3)

public void Test2() {

System.out.println("TestA");

Assert.assertEquals("Exp", "Exp1");

}

@Test (priority = 2)

public void Test3() {

System.out.println("TestK");

}

}

**Example With dependsOnMethod and alwaysRun=true/false options**

package testPackage1;

import org.testng.Assert;

import org.testng.annotations.Test;

public class TestNG {

@Test

public void Test1() {

System.out.println("TestF");

Assert.assertEquals("Exp1", "Exp");

}

@Test (dependsOnMethods={"Test1"}, alwaysRun=true)

public void Test2() {

System.out.println("TestA");

Assert.assertEquals("Exp", "Exp1");

}

@Test (priority = 2)

public void Test3() {

System.out.println("TestK");

}

}

**Example with @BeforeMethod and @AfterMethod Annotations**

package testPackage1;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class TestNG {

@BeforeMethod

public void Login() {

System.out.println("Login Successful");

}

@Test (priority = 1)

public void SearchProduct() {

System.out.println("Search Product Successful");

}

@Test (priority = 2)

public void CreateOrder() {

System.out.println("Create Order Done");

}

@Test (priority = 3)

public void Payment() {

System.out.println("Payment Successful");

}

@AfterMethod

public void Logout() {

System.out.println("Logout Successful");

}

}

**@BeforeClass and @AfterClass : These annotations will execute once per class, that is before executing all the test of a class and after running all the tests of a class.**

**Example Using @BeforeClass and @AfterClass Annotations**

package testPackage1;

import org.testng.annotations.AfterClass;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class TestNG {

@BeforeClass

public void Login() {

System.out.println("Login Successful");

}

@Test (priority = 1)

public void SearchProduct() {

System.out.println("Search Product Successful");

}

@Test (priority = 2)

public void CreateOrder() {

System.out.println("Create Order Done");

}

@Test (priority = 3)

public void Payment() {

System.out.println("Payment Successful");

}

@AfterClass

public void Logout() {

System.out.println("Logout Successful");

}

}

**@BeforeTest and @AfterTest : These will run before any test from multiple classes and after test from multiple classes respectively.**

**Example :**

1. Create two TestNG Classes TestNgTest1 and TestNgTest2
2. Have few @Test in TestNgTest1 and few more in TestNG2
3. Use @BeforeTest for Login in one of the TestNG Test.
4. Use @AfterTest for Logout in one of the TestNG Test.
5. Create a TestNG.xml for running both the TestNG Classes basically the entire suite.

<?xml version="1.0" encoding="UTF-8"?>

<suite name="Suite" >

<test name="Test">

<classes>

<class name="testPackage1.TestNG"/>

<class name="testPackage1.TestNG2"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite →

**@BeforeSuite and @AfterSuite : Used only once before a Test Suite and After a Test Suite**

**Example with Actitime Application**

package com.testng;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.Test;

public class ActiTimeLogin {

public static WebDriver driver;

@BeforeClass

public void setUp() {

driver = new FirefoxDriver();

driver.get("http://demo.actitime.com/login.do");

driver.manage().window().maximize();

}

@Test

public void loginTest() {

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

WebElement objUserName = driver.findElement(By.name("username"));

if (objUserName.isDisplayed()) {

System.out

.println("Username element found in Application and Script executing ....");

if (objUserName.isEnabled()) {

System.out.println("Username element is enable for typing");

objUserName.sendKeys("user");

driver.findElement(By.name("pwd")).sendKeys("user");

driver.findElement(By.xpath("//a[@id='loginButton']/div")).click();

driver.findElement(By.linkText("Logout")).click();

} else {

System.out.println("Username element is disable for typing");

}

}

else {

System.out.println("Username element not found in Application");

}

}

@AfterClass

public void tearDown() {

driver.quit();

}

}

**testng.xml**

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name="TrainingTestSuite" annotations= "JDK" thread-count="1">

<test name="Testing login on Actitime Application">

<classes>

<class name="com.testng.ActiTimeLogin" />

</classes>

</test>

</suite>

**Example on Groupings like P1, P2 Tests or BVT, Regression Tests**

package testpackage1;

import org.testng.Assert;

import org.testng.annotations.Test;

public class TestNGTest1 {

@Test(groups = {"p1"})

public void Test1() {

System.out.println("Test f Function");

Assert.assertEquals("Exp", "Exp");

}

@Test (groups = {"BVT"})

public void Test2() {

System.out.println("TestA");

Assert.assertEquals("Exp", "Exp");

}

@Test(groups = {"p1"})

public void Test3() {

System.out.println("Test k Function");

}

}

**TestNG XML**

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name="TestSuite">

<test name="P1 Tests">

<groups>

<run>

<include name="BVT">

</include>

</run>

</groups>

<classes>

<class name="testpackage1.TestNGTest1" />

</classes>

</test>

</suite>

**TestNG Parameters Example**

package testPackage1;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

public class ParametersExample {

private static WebDriver driver;

@Test

@Parameters({ "sUsername", "sPassword" })

public void test(String sUsername, String sPassword) {

driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

driver.get("http://www.slideshare.net");

driver.findElement(By.linkText("Login")).click();

driver.findElement(By.name("user\_login")).sendKeys(sUsername);

driver.findElement(By.id("user\_password")).sendKeys(sPassword);

driver.findElement(By.id("login\_from\_loginpage")).click();

driver.close(); }

}

**TestNG.xml**

<?xml version="1.0" encoding="UTF-8"?>

<suite name="Suite" parallel="false">

<test name="Test">

<parameter name="sUsername" value="test@gmail.com"/>

<parameter name="sPassword" value="Test123"/>

<classes>

<class name="testPackage1.ParametersExample"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

## **Example With Parameters to run on Multiple Browsers and Reporter Logging**

## package testpackage1;

## import java.util.concurrent.TimeUnit;

## import org.openqa.selenium.By;

## import org.openqa.selenium.WebDriver;

## import org.openqa.selenium.WebElement;

## import org.openqa.selenium.chrome.ChromeDriver;

## import org.openqa.selenium.firefox.FirefoxDriver;

## import org.openqa.selenium.ie.InternetExplorerDriver;

## import org.testng.Reporter;

## import org.testng.annotations.AfterClass;

## import org.testng.annotations.BeforeClass;

## import org.testng.annotations.Parameters;

## import org.testng.annotations.Test;

## public class MultipleBrowsers {

## public static WebDriver driver;

## @BeforeClass

## @Parameters({ "url", "browserName" })

## public void setUp(String url, String browserName) {

## if (browserName.equals("IExplore")) {

## System.setProperty("webdriver.ie.driver", "browsers//IEDriverServer.exe");

## driver = new InternetExplorerDriver();

## Reporter.log("Script execting on " + browserName + "browser", true);

## } else if (browserName.equals("Firefox")) {

## driver = new FirefoxDriver();

## Reporter.log("Script execting on " + browserName + "browser", true);

## } else if (browserName.equals("Chrome")) {

## System.setProperty("webdriver.chrome.driver", "browsers//chromedriver.exe");

## driver = new ChromeDriver();

## Reporter.log("Script execting on " + browserName + "browser", true);

## } else {

## driver = new FirefoxDriver();

## Reporter.log("Script execting on default firefox browser", true);

## }

## driver.get(url);

## driver.manage().window().maximize();

## }

## @Test

## @Parameters({ "userName", "password" })

## public void loginTest(String userName, String password) {

## driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

## WebElement objUserName = driver.findElement(By.name("username"));

## if (objUserName.isDisplayed()) {

## Reporter.log("Username element found in Application and Script executing ....",true);

## if (objUserName.isEnabled()) {

## Reporter.log("Username element is enable for typing", true);

## objUserName.sendKeys(userName);

## driver.findElement(By.name("pwd")).sendKeys(password);

## driver.findElement(By.xpath("//a[@id='loginButton']/div")).click();

## driver.findElement(By.linkText("Logout")).click();

## } else {

## Reporter.log("Username element is disable for typing", true);

## }

## }

## else {

## Reporter.log("Username element not found in Application", true);

## }

## }

## @AfterClass

## public void tearDown() {

## driver.close();

## }

## }

**TestNg.xml**

## <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

## <suite name="TestNGTrainingTestSuite" annotations="JDK" thread-count="1">

## <parameter name="url" value="http://demo.actitime.com/login.do"></parameter>

## <parameter name="userName" value="user"></parameter>

## <parameter name="password" value="user"></parameter>

## <test name="Testing login on Actitime Application using Firefox">

## <parameter name="browserName" value="Firefox"></parameter>

## <classes>

## <class name="testpackage1.MultipleBrowsers" />

## </classes>

## </test>

## <test name="Testing login on Actitime Application using IExplore">

## <parameter name="browserName" value="IExplore"></parameter>

## <classes>

## <class name="testpackage1.MultipleBrowsers" />

## </classes>

## </test>

## <test name="Testing login on Actitime Application using Chrome">

## <parameter name="browserName" value="Chrome"></parameter>

## <classes>

## <class name="testpackage1.MultipleBrowsers" />

## </classes>

## </test>

## </suite>

## **TestNG Data Providers**

When you need to pass complex parameters or parameters that need to be created from Java (complex objects, objects read from a property file or a database, etc…), in such cases parameters can be passed using Data providers. A Data Provider is a method annotated with *@DataProvider*. A Data Provider returns an array of objects.

**Basic Example to Understand the DataProvider Annotation**

**package testpackage1;**

**import org.testng.annotations.DataProvider;**

**import org.testng.annotations.Test;**

**public class DataParameterization {**

**@Test(dataProvider = "getData")**

**public void LoginTest(String UserName, String Passwd){**

**System.out.println(UserName);**

**System.out.println("Passwd");**

**}**

**@DataProvider**

**public Object[][] getData(){**

**Object[][] data = new Object[2][2];**

**data[0][0]= "shanker";**

**data[0][1]= "pwd123";**

**data[1][0]="test";**

**data[1][1]="pwd123";**

**return data;**

**}**

**}**

package testPackage1;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

public class ParametersExample {

private static WebDriver driver;

@DataProvider(name = "Authentication")

public static Object[][] Credentials() {

return new Object[][] { { "testuser1", "Test@123" }, { "testuser2", "Test@123" }};

}

@Test(dataProvider = "Authentication")

public void test(String sUsername, String sPassword) {

driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

driver.get("http://www.slideshare.net");

driver.findElement(By.linkText("Login")).click();

driver.findElement(By.name("user\_login")).sendKeys(sUsername);

driver.findElement(By.id("user\_password")).sendKeys(sPassword);

driver.findElement(By.id("login\_from\_loginpage")).click();

driver.close();

}

}

**TestNg.XML**

<?xml version="1.0" encoding="UTF-8"?>

<suite name="Suite" parallel="false">

<test name="Test">

<classes>

<class name="testPackage1.ParametersExample"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**Script to Read Data From ExcelSheet (97-2003 format of Excel)**

package testpackage1;

import java.io.File;

import java.io.FileInputStream;

import org.apache.poi.hssf.usermodel.HSSFCell;

import org.apache.poi.hssf.usermodel.HSSFRow;

import org.apache.poi.hssf.usermodel.HSSFSheet;

import org.apache.poi.hssf.usermodel.HSSFWorkbook;

public class ExcelReadTest {

public static void main(String[] args) throws Exception {

// TODO Auto-generated method stub

File excel = new File("D:\\EclipseWS\\TestData.xls");

FileInputStream fis = new FileInputStream(excel);

HSSFWorkbook wb = new HSSFWorkbook(fis);

HSSFSheet ws = wb.getSheet("Sheet1");

int rowNum = ws.getLastRowNum() + 1;

int colNum = ws.getRow(0).getLastCellNum();

System.out.println(rowNum);

System.out.println(colNum);

String[][] data = new String[rowNum][colNum];

for (int i=0; i < rowNum; i++){

HSSFRow row = ws.getRow(i);

for (int j=0; j < colNum; j++){

HSSFCell cell = row.getCell(j);

String Value = cellToString(cell);

data[i][j] = Value;

System.out.println("the value is" + Value);

}

}

}

public static String cellToString(HSSFCell cell){

int type;

Object result;

type = cell.getCellType();

switch (type) {

case 0 : //numeric value in excel

result = cell.getNumericCellValue();

break;

case 1 : //String Value in Excel

result = cell.getStringCellValue();

break;

default :

throw new RuntimeException("There is no support for this type of value in Apche POI");

}

return result.toString();

}

}

**/\*String[] A = new String[5];**

**A[0] = "QEdge";**

**A[1] = "Ammerpet";**

**A[2] = "Hyderabad";**

**for(int i=0; i<A.length; i++)**

**{**

**System.out.println(A[i]);**

**}\*/**

**Object B[][] = new Object[2][3];**

**B[0][0] = 10;**

**B[0][1] = "QEdge";**

**B[1][0] = "Ammerpet";**

**B[1][1] = "hyderabad";**

**System.out.println(B[0][0]);**

**System.out.println(B[0][1]);**

**System.out.println(B[1][0]);**

**System.out.println(B[1][1]);**

**Script to Read the Data from Excel with latest format (.xlsx)**

package testpackage1;

import java.io.File;

import java.io.FileInputStream;

import org.apache.poi.xssf.usermodel.XSSFCell;

import org.apache.poi.xssf.usermodel.XSSFRow;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ExcelReadTest {

public static void main(String[] args) throws Exception {

// TODO Auto-generated method stub

File excel = new File("D:\\EclipseWS\\TestData1.xlsx");

FileInputStream fis = new FileInputStream(excel);

XSSFWorkbook wb = new XSSFWorkbook(fis);

XSSFSheet ws = wb.getSheet("Sheet1");

int rowNum = ws.getLastRowNum() + 1;

int colNum = ws.getRow(0).getLastCellNum();

System.out.println(rowNum);

System.out.println(colNum);

String[][] data = new String[rowNum][colNum];

for (int i=0; i < rowNum; i++){

XSSFRow row = ws.getRow(i);

for (int j=0; j < colNum; j++){

XSSFCell cell = row.getCell(j);

String Value = cellToString(cell);

data[i][j] = Value;

System.out.println("the value is" + Value);

}

}

}

public static String cellToString(XSSFCell cell){

int type;

Object result;

type = cell.getCellType();

switch (type) {

case 0 : //numeric value in excel

result = cell.getNumericCellValue();

break;

case 1 : //String Value in Excel

result = cell.getStringCellValue();

break;

default :

throw new RuntimeException("There is no support for this type of value in Apche POI");

}

return result.toString();

}

}

**Data Driven Test for the Login form with multiple userids and passwords**

package testPackage1;

import java.io.File;

import java.io.FileInputStream;

import java.util.concurrent.TimeUnit;

import org.apache.poi.xssf.usermodel.XSSFCell;

import org.apache.poi.xssf.usermodel.XSSFRow;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

public class SlideShareDDTTest1 {

@Test(dataProvider = "getData")

public void LoginTest(String UserName, String Passwd){

WebDriver driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

driver.get("http://www.slideshare.net");

driver.findElement(By.linkText("Login")).click();

driver.findElement(By.name("user\_login")).sendKeys(UserName);

driver.findElement(By.id("user\_password")).sendKeys(Passwd);

driver.findElement(By.id("login\_from\_loginpage")).click();

driver.close();

}

@DataProvider

public String[][] getData() throws Exception{

File excel = new File("D:\\Selenium\\WorkingFolder\\TestData.xlsx");

FileInputStream fis = new FileInputStream(excel);

XSSFWorkbook wb = new XSSFWorkbook(fis);

XSSFSheet ws = wb.getSheet("Sheet1");

int rowNum = ws.getLastRowNum() + 1;

int colNum = ws.getRow(0).getLastCellNum();

String[][] data = new String[rowNum][colNum];

for (int i=1; i < rowNum; i++){

XSSFRow row = ws.getRow(i);

for (int j=0; j < colNum; j++){

XSSFCell cell = row.getCell(j);

String Value = cellToString(cell);

data[i][j] = Value;

}

}

return data;

}

public static String cellToString(XSSFCell cell){

int type;

Object result;

type = cell.getCellType();

switch (type) {

case 0 : //numeric value in excel

result = cell.getNumericCellValue();

break;

case 1 : //String Value in Excel

result = cell.getStringCellValue();

break;

default :

throw new RuntimeException("There is no support for this type of value in Apche POI");

}

return result.toString();

}

}